

Appl. Ser. No. 10/573,342
Amendment dated December 21, 2007
Reply to Office Action mailed September 26, 2007

Amendments to the Specification

Please add the following Code Appendix, which was missing from the published application U.S. Pub. No. US 2007/0023444, but incorporated by reference in paragraph 33 of the published application and included with the International PCT Application:

CODE APPENDIX

```
; ; Assembly code for PillSafe
;

list p=12f675
include "p12f675.inc"

timer_cnt    equ    0x20
isr_w_save   equ    0x21
isr_status_save    equ    0x22
WAIT_H        equ    0x23
WAIT_L        equ    0x24
CNT1          equ    0x25
CNT0          equ    0x26

GP_SOLENOID  equ    0x0
GP_BUTTON_USER    equ    0x1
GP_LED         equ    0x2
GP_BUTTON_DONE  equ    0x3

__CONFIG      _CPD_OFF & _CP_OFF & _BODEN_OFF & _MCLRE_OFF & _PWRTE_OFF &
_WDT_OFF & _INTRC_OSC_NOCLKOUT

goto main

org 0x004
dispatch_interrupt:
```

Appl. Ser. No. 10/573,342
Amendment dated December 21, 2007
Reply to Office Action mailed September 26, 2007

```
; ; save W, STATUS
movwf isr_w_save
swapf STATUS, W      ; swapf does not affect status reg.
movwf isr_status_save

btfsr PIR1, TMR1IF      ; did we get here because of a timer1 overflow?
call timer1_isr
btfsr INTCON, GPIF      ; interrupt on GPIO pin?
call gpio_change_isr

; ; restore W, Status
swapf isr_status_save, W
movwf STATUS
swapf isr_w_save, F      ; swapf does not affect STATUS
swapf isr_w_save, W
retfie

timer1_isr:
    ; ; clear timer interrupt flag, and set timer_cnt flag
    bcf PIR1, TMR1IF
;    bsf timer_cnt, 0
    return

gpio_change_isr:
    ; ; read from GPIO to prevent GPIF getting set again, and clear GPIF
    movf GPIO, F
    bcf INTCON, GPIF
    return

; ; main

main:
    bcf STATUS, RP0
    clrf GPIO
    movlw 0x7
    movwf CMCON      ; disable comparator
    clrf TMRO
```

Appl. Ser. No. 10/573,342
Amendment dated December 21, 2007
Reply to Office Action mailed September 26, 2007

```
        movlw 0x40      ; enable peripheral interrupts
        movwf INTCON
        clrf T1CON      ; timer1 off
        clrf TMR1L      ; clear timer1
        clrf TMR1H
        clrf PIR1
        clrf ADCONO
        ;; ****
        ;; BANK1
        ;; ****
        bsf STATUS, RP0
        movlw ~((1 << GP_SOLENOID) | (1 << GP_LED))
        movwf TRISIO
        clrf VRCON
        clrf OPTION_REG ; enable weak pull-ups
        clrf WPU         ; use pull-ups with buttons
        clrf ANSEL
        movlw 0x01      ; enable timer1 interrupt
        movwf PIE1
        bcf STATUS, RP0
        ;; ****
        ;; BANK0
        ;; ****
        bsf INTCON, GIE ; enable all unmasked interrupts
infinite:
        movlw 0x03      ; WAIT = 0x0203
        movwf WAIT_L
        movlw 0x02
        movwf WAIT_H
        call wait_long
        bsf GPIO, GP_LED
        call wait_for_button
        bcf GPIO, GP_LED
        call dispense
        goto infinite
        ;; ****
```

Appl. Ser. No. 10/573,342
Amendment dated December 21, 2007
Reply to Office Action mailed September 26, 2007

```
; ; wait_for_timeout - sleep until the desired time has passed expires
; ; ****
; ; TODO: longer delays, sleep wait
wait_for_timeout:
    ;; setup timer interrupt
    clrf T1CON      ; timer1 off
    clrf TMR1L      ; clear timer1
    clrf TMR1H
    movlw 0x0f      ; timer1 always on, prescale 8:1
                    ; LP oscillator, async mode, timer1 on
    movwf T1CON
    bcf PIR1, TMR1IF
    bsf STATUS, RP0 ; *** bank1
    bsf PIE1, TMR1IE ; enable timer1 interrupt
    bcf STATUS, RP0 ; *** bank0

    ; sleep (or wait) repeatedly until timeout period is over
    sleep
;     bcf timer_cnt, 0
; t_wait: btfss timer_cnt, 0
;     goto t_wait

    ;; Disable timer and timer interrupt
    bsf STATUS, RP0 ; *** bank1
    bcf PIE1, TMR1IE
    bcf STATUS, RP0 ; *** bank0
    return

; ; ****
; ; wait_long - Decrement WAIT_L to 0 WAIT_H times with prescale set
; ;          to 8:1.
; ; When WAIT_L is 15, this WAIT_H will be the number of 4
; ; minute intervals. When WAIT_L is 225, WAIT_H is the number
; ; of hours to wait.
; ; ASSUME: WAIT_H and WAIT_L are both at least 1
; ; ****
wait_long:
```

Appl. Ser. No. 10/573,342
Amendment dated December 21, 2007
Reply to Office Action mailed September 26, 2007

```
    clrf T1CON      ; turn timer1 off
    ;; setup tmr1h and tmr1l
    clrf TMR1L
    clrf TMR1H
    movlw ((1<<T1CKPS1) | (1<<T1CKPS0) | (1<<T1OSCEN) | (1<<NOT_T1SYNC) | (1<<TMR1CS
) | (1<<TMR1ON))
    movwf T1CON
    movf WAIT_H, W
    movwf CNT1
wait_long_loop_h:      ; do {
    movf WAIT_L, W
    movwf CNT0
wait_long_loop_l:      ; do {
    call wait_for_timer1
    decfsz CNT0, F          ; } while(CNT0 > 0);
    goto wait_long_loop_l
    decfsz CNT1, F          ; } while(CNT1 > 0);
    goto wait_long_loop_h
    bcf T1CON, TMR1ON
    return

    ;; ****
    ;; wait_ticks - sleep for number of timer1 ticks in WAIT_H, WAIT_L
    ;; ****
wait_ticks:
    clrf T1CON      ; turn timer1 off, prescaling to 1:1
    ;; setup tmr1h and tmr1l
    comf WAIT_L, W
    movwf TMR1L
    comf WAIT_H, W
    movwf TMR1H
    incfsz TMR1L, F
    decf TMR1H, F
    incf TMR1H, F

    movlw ((1<<T1OSCEN) | (1<<NOT_T1SYNC) | (1<<TMR1CS) | (1<<TMR1ON))
    movwf T1CON
```

Appl. Ser. No. 10/573,342
Amendment dated December 21, 2007
Reply to Office Action mailed September 26, 2007

```
call  wait_for_timer1
bcf   T1CON, TMR1ON
return

;;; *****
;;; wait_for_timer1 - sleep until timer1 interrupts
;;
;;; ASSUME: TMR1L, TMR1H, and prescaling bits are already set
;;; ASSUME: The value in TMR1H & TMR1L is big enough that timer1
;;;           will not interrupt before wait_for_timer1 sleeps
;;; ASSUME: No extraneous interrupts will wake wait_for_timer1 from sleep
;;; *****
;;; TODO: longer delays, sleep wait
wait_for_timer1:
    bcf   PIR1, TMR1IF
    ;; TODO: maybe leave timer1 interrupt enabled all the time.
    ;; As long as timer is off, no interrupts will happen
    bsf   STATUS, RPO ; *** bank1
    bsf   PIE1, TMR1IE      ; enable timer1 interrupt
    bcf   STATUS, RPO ; *** bank0

    ;; sleep repeatedly until timeout period is over
    sleep

    ;; Disable timer and timer interrupt
    bsf   STATUS, RPO ; *** bank1
    bcf   PIE1, TMR1IE
    bcf   STATUS, RPO ; *** bank0
    return

;; *****
;;; wait_for_button - sleep until a button is pressed
;;; *****
wait_for_button:
    ;; setup button interrupt
    bsf   STATUS, RPO ; *** bank1
    bsf   WPU, GP_BUTTON_USER
    bsf   IOC, GP_BUTTON_USER
```

Appl. Ser. No. 10/573,342

Amendment dated December 21, 2007

Reply to Office Action mailed September 26, 2007

```
bcf STATUS, RP0 ; *** bank0
bsf INTCON, GPIE

; sleep until button interrupt
; TODO: is polling the button necessary, or even a good thing?
sleep_wait:
    sleep
    btfsc GPIO, GP_BUTTON_USER
    goto sleep_wait

; disable button interrupt
bcf INTCON, GPIE
bsf STATUS, RP0 ; *** bank1
bcf IOC, GP_BUTTON_USER
bcf WPU, GP_BUTTON_USER
bcf STATUS, RP0 ; *** bank0
return
;*****dispense - dispense a pill (activate the solenoid)
;*****dispense:
dispense:
    ; enable solenoid and sleep until it has moved (use 100 ms)

    bsf GPIO, GP_SOLENOID ; GP_SOLENOID = 1
    movlw 0xcd          ; WAIT = 0x0ccd
    movwf WAIT_L
    movlw 0x0c
    movwf WAIT_H
    call wait_ticks ; wait_ticks()
    bcf GPIO, GP_SOLENOID ; GP_SOLENOID = 0
    return
end
```